**Journal: Reflection**

Justin Starr

Department of STEM

CS 305 – Software Security

Dr. Sarah North

September 17, 2023

**Journal: Reflection**

After reviewing the article, *DevSecOps: A Systematic Approach for Secure Software Development* (Jeganathan, 2019), the following analysis was conducted as a reflection on the importance of software security.

* What is your role in solving security concerns as a developer? What might solving security concerns as a developer involve?

As a software developer, it is paramount to be thinking about security. As a developer, whether developing applications or entire systems, I need to think about potential vulnerabilities and how I can be coding in a more secure way to help prevent vulnerabilities. Secure coding is a developmental practice meaning only if I employ secure coding techniques will I be able to produce secure code. Solving security concerns as a developer involves staying informed about vulnerabilities and exploits, as well as following established threat models both simple and complex (Jeganathan, S., 2019). When coding applications this can include “applying controls for input validation, session management, user credential validation, user access control, data protection and privacy, logging, API security, detecting security misconfigurations, etc.” (Jeganathan, S., 2019). No matter what is being worked on, security should be involved at every level.

* Where does security fall within the software stack and development life cycle?

Security is often looked at as an afterthought, where much of the time, security testing is often conducted during specific testing phases of the software development life cycle (Jeganathan, S., 2019). Fortunately, DevSecOps, which is a DevOps methodology, seeks to incorporate security as early as possible in the software development life cycle (Jeganathan, S., 2019). I believe that security should always be at the forefront of the developmental process and be incorporated into every phase possible, from requirements gathering, to design, testing, and production. In every phase, no matter what our role is, we should be considering what security risks could be associated.

* How might you add security measures to transform a DevOps pipeline into a DevSecOps pipeline?

DevSecOps can be integrated with a DevOps pipeline by integrating a Continuous Security pipeline that runs concurrently with the DevOps pipeline. There is no one perfect solution for doing so, however, one could potentially be modeled as follows:

Development Starts with the DevOps Team and then phases to Continuous Integration, to Continuous Delivery, to a live environment, and then back to the DevOps Team (Jeganathan, S., 2019). Continuous Security runs simultaneously with secure design and engineering taking place with the DevOps Team, Security Testing occurring alongside continuous integration and continuous delivery, security monitoring occurring in the live environment, and security risk management before the project phases back to the DevOps team (Jeganathan, S., 2019). This way, security has a place in all aspects of project development.

* The article suggests creating and following a plan to secure the entire DevOps life cycle. What is included in the suggested plan, and would you recommend following it?

The plan suggests that there are four key phases of continuous security that happen during the SDLC, security design and engineering, security testing, security monitoring, and security risk management. These four phases should run concurrently with the DevOps model. During the planning stage, security requirements and design are completed for the software development and for securing the DevOps life cycle entirely (Jeganathan, S., 2019). In the coding phase, secure coding practices and enforcing industry followed secure-coding practices are implemented along with training developers to practice security principles, peer review is conducted, the design of unit tests scrips that have an emphasis on security, and the elimination of vulnerable components (Jeganathan, S., 2019). Once coding has finished, the build stage begins where all of the software modules are made available to the source code repository where it can then be made ready for deployment. During this phase, security is maintained by implementing various practices such as applying access and separation of duties controls, adopting industry-followed automation tools, adopting native access controls, and even applying two-factor authentication to protect repositories (Jeganathan, S., 2019). Once building has completed, the project can move on to the testing phase where tests can be created with a focus on security alongside the functionality of the software being developed. Once the product has been tested, it can move to the release and deploy phase where security still plays a vital role in ensuring that proper authentication controls have been applied, user access controls are in place, privileged credentials are protected, and container security best practices are applied (Jeganathan, S., 2019). Lastly, the project moves to the operating phase where the security team is “actively involved in the continuous monitoring, analysis, and protection for the live-run environment.”

By following this plan, we can see how DevSecOps runs parallel with DevOps. Security is implemented in every phase of the software development life cycle and is not given the chance to be an afterthought. I do believe that incorporating security into the software development life cycle is appropriate and I would strongly recommend that, when possible, this type of model or plan should be followed. The result will be more robust, secure software.

**References**

Jeganathan, S. (2019). DevSecOps: A Systemic Approach for Secure Software

Development. *ISSA Journal*, 17(11), 20-27. <https://eds-s-ebscohost-com.ezproxy.snhu.edu/eds/detail/detail?vid=0&sid=9b2e0ac9-fb67-4ed8-8b2b-521579e263a8%40redis&bdata=JnNpdGU9ZWRzLWxpdmUmc2NvcGU9c2l0ZQ%3d%3d#AN=139557317&db=tsh>